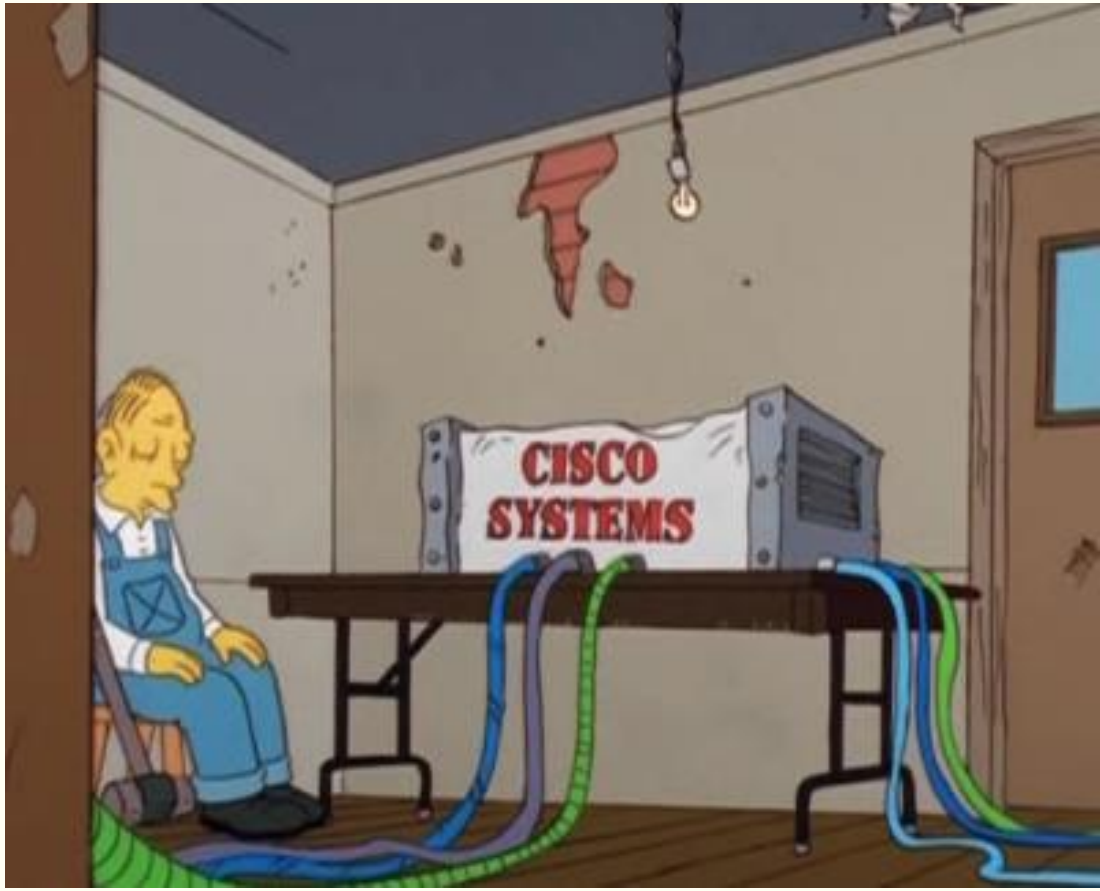
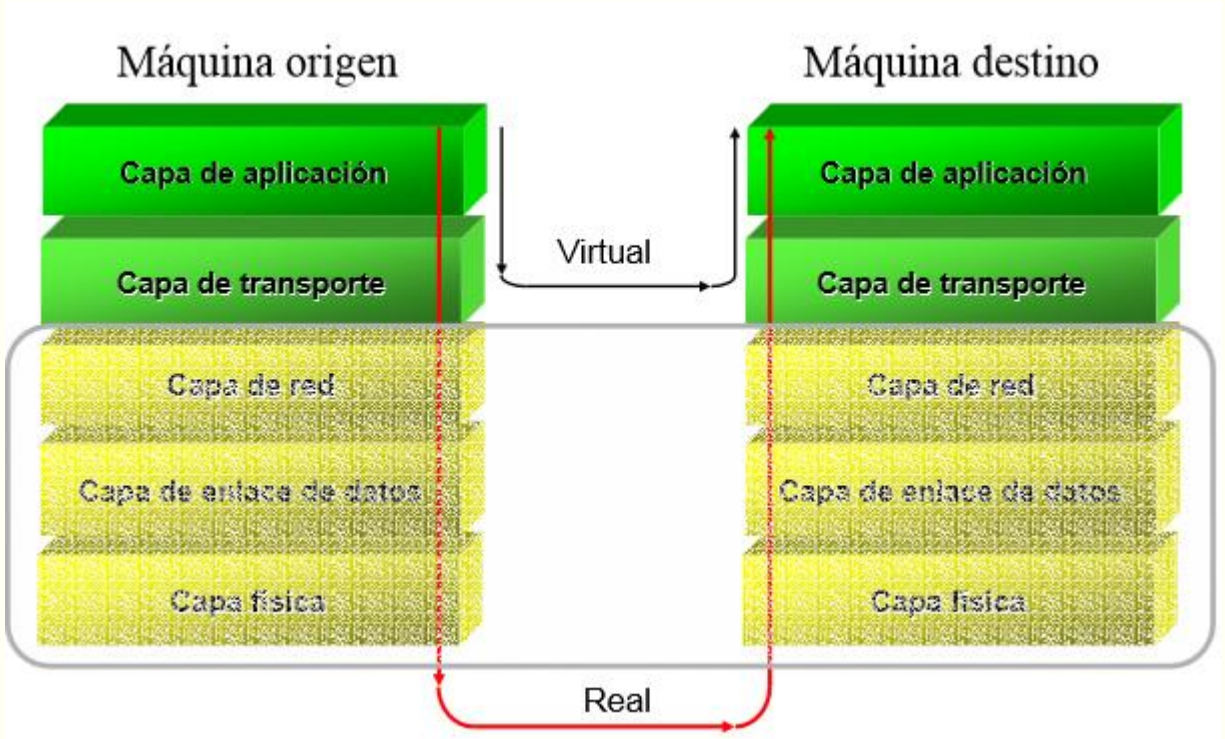
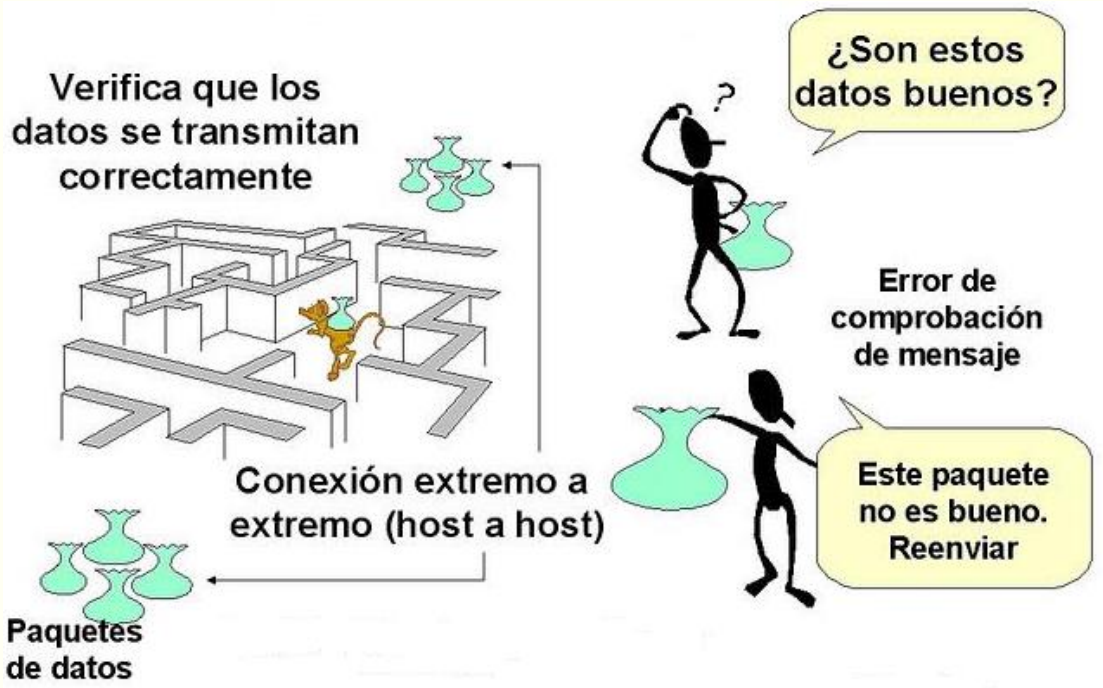
CAPA DE TRANSPORTE



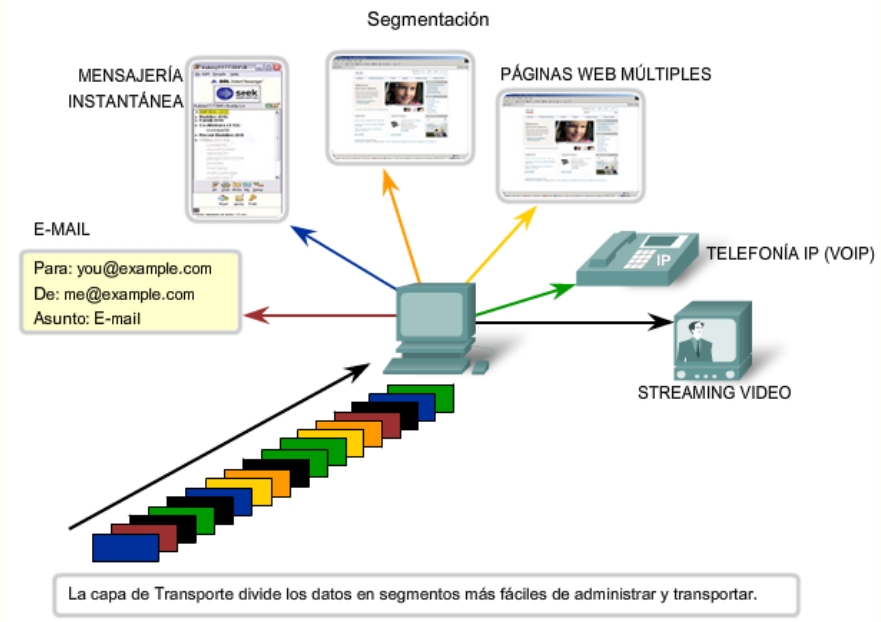
Comunicación





Funciones

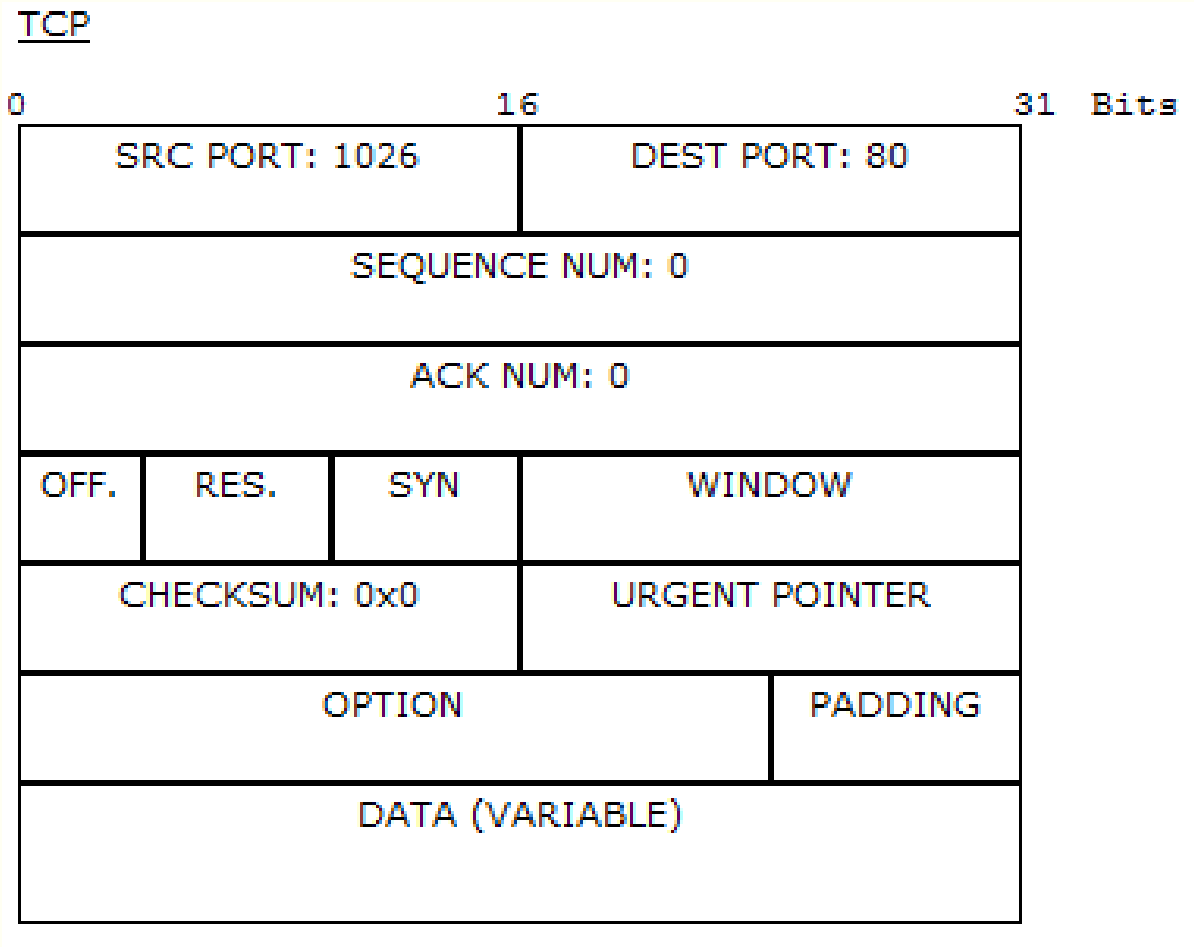
- Administrar los datos de aplicación para las conversaciones entre hosts.
- Se encarga de generar la comunicación entre host origen y destino.
- Segmenta la información.
- La segmentación da lugar a la multiplexión en la cual múltiples sesiones pueden utilizar la misma red enviando o recibiendo datos al mismo tiempo.



Características de TCP y UDP

TCP	UDP
Orientado a la conexión	Sin conexión
Confiabilidad en la entrega de mensajes	No se fragmentan los mensajes
Divide los mensajes en datagramas	No hay reensamblaje ni sincronización
Hace seguimiento del orden (o secuencia)	En caso de error, el mensaje se retransmite
Usa checksums para la detección de errores	Sin acuse de envío
La confiabilidad es prioridad	Los mensajes del servidor y el cliente entran
Los mensajes exceden el tamaño de un paquete	completamente dentro de un paquete
UDP	El servidor maneja múltiples clientes
RFC 793	RFC 768

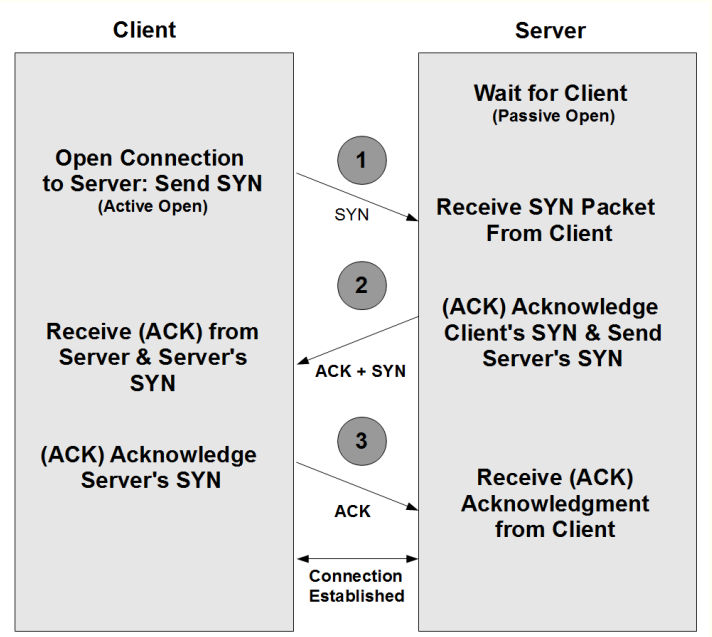
Segmento TCP



Conexión

- Establece las conexiones usando el protocolo de acuerdo a tres vías (three-way handshake).

Para establecer una conexión:



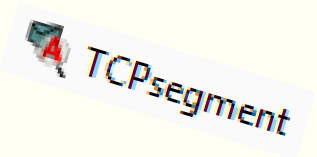
Servidor

- Ejecuta las primitivas LISTEN y ACCEPT

Cliente

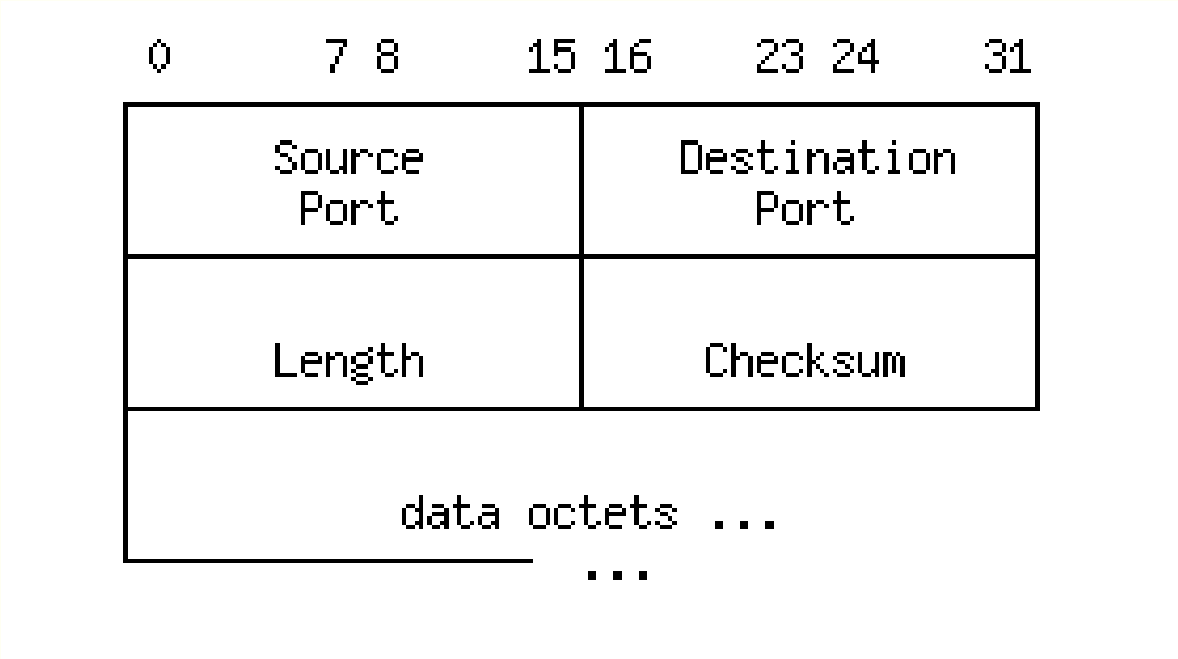
- ejecuta una primitiva CONNECT especifica:
  - la dirección y el puerto IP con el que se desea conectar,
  - el tamaño máximo de segmento TCP que está dispuesto a aceptar y opcionalmente algunos datos de usuario (ejemplo: contraseña).
  - envía un segmento TCP con el bit SYN encendido y el bit ACK apagado, y espera una respuesta.

Estados



- **CLOSED:** No hay conexión activa ni pendiente.
- **LISTEN:** El servidor espera una llamada.
- **SYN RCVD:** Llegó una solicitud de conexión; espera ACK.
- **SYN SENT:** La aplicación comenzó a abrir una conexión.
- **ESTABLISHED:** Estado normal de transferencia de datos.
- **FIN WAIT 1:** La aplicación dijo que ya terminó.
- **FIN WAIT 2:** El otro lado acordó liberar.
- **TIMED WAIT:** Espera a que todos los paquetes mueran.
- **CLOSING:** Ambos lados intentaron cerrar simultáneamente.
- **CLOSE WAIT:** El otro lado inició una liberación.
- **LAST ACK:** Espera a que todos los paquetes mueran.

Segmento UDP



UDPsegment

IANA

www.iana.org [RFC 3232]

- Internet Assigned Numbers Authority.
- Asigna los números de puerto.
- Tipos de números de puerto:
- **Puertos Bien Conocidos** (Well Known Ports) (0-1023): reservados para servicios y aplicaciones.
- **Puertos Registrados** (Registered Ports) (1024-49151): son asignados para usuarios de procesos o aplicaciones.
- **Puertos Dinámicos o Privados** (Dynamic or Private) 49152-65535): son asignados a los clientes cuando inician una conexión.

http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

Ejemplos puertos TCP

Puerto	Servicio o aplicación
21	FTP
23	Telnet
25	SMTP
63	Whois
70	Gopher
79	Finger
80	HTTP
110	POP3
119	NNTP

Ejemplos puertos UDP

Puerto	Servicio o aplicación
69	TFTP
520	RIP

Ejemplos puertos TCP/UDP

Puerto	Servicio o aplicación
53	DNS
161	SNMP

NETSTAT

- Brinda información sobre las conexiones establecidas por el equipo como
  - ✓ puertos abiertos,
  - ✓ conexiones en segundo plano,
  - ✓ conexiones establecidas por programas espía

COMANDOS

- **netstat**
  - Muestra las conexiones activas recientes
- **netstat –an**
  - Muestra todas las conexiones y los puertos abiertos
- **netstat –o**
  - Muestra la identidad del proceso de cada conexión
- **netstat –b**
  - Muestra el nombre del archivo que inició la conexión ya sea un programa o un virus existente en el equipo.

CHECKSUM

Detectar “errores” (bits cambiados) en segmentos transmitidos.

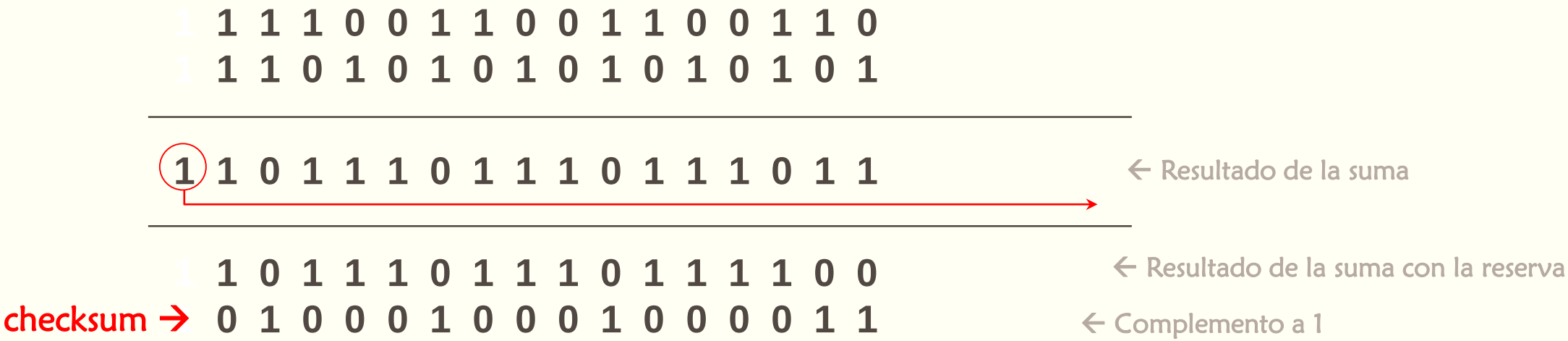
Transmisor	Receptor

• Trata el contenido de cada segmento como una secuencia de enteros de 16 bits.
• checksum: suma del contenido del segmento y luego tomamos el complemento 1.
• Transmisor pone el valor del checksum en el campo checksum del datagrama UDP.
	• Calcula el checksum del segmento recibido.
	• Verifica si el valor calculado **corresponde** al valor de checksum recibido en el campo:
	  • NO → error detectado
	  • SI → no hay error detectado

Ejemplo

- Notar
  - Cuando sumamos números, la reserva del bit más significativo debe ser sumada al resultado.
  - Tomar el complemento 1 → invertir los bits: 0's → 1

    1's → 0
- Ejemplo: sumar dos enteros de 16-bits



Bibliografía

- *Computer Networking: A Top Down Approach*
  4th edition
  Jim Kurose, Keith Ross
  Addison-Wesley, July 2007, ISBN: 9780321497703

- *CCNA R&S: Introduction to Networks*
  Capítulo 7